ARL-TR-7532 ● SEP 2015

US Army Research Laboratory

Characterization of Extremely Lightweight Intrusion Detection (ELIDe) Power Utilization with Varying Throughput and Payload Sizes

by Garrett S Payer, Ken F Yu, and Richard Harang

NOTICES

Disclaimers

ARL-TR-7532 ● SEP 2015

US Army Research Laboratory

Characterization of Extremely Lightweight Intrusion Detection (ELIDe) Power Utilization with Varying Throughput and Payload Sizes

by Garrett S Payer, Ken F Yu, and Richard E Harang
Computational and Information Sciences Directorate, ARL

REPORT DOCUMENTATION PAGE			*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* Sep 2015	**2. REPORT TYPE** Final	**3. DATES COVERED (From - To)** Apr– Sep 2014

4. TITLE AND SUBTITLE Characterization of Extremely Lightweight Intrusion Detection (ELIDe) Power Utilization with Varying Throughput and Payload Sizes	**5a. CONTRACT NUMBER** W911QX-07-F-0020
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Garrett S Payer, Ken F Yu, and Richard E Harang	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) US Army Research Laboratory ATTN: RDRL-CIN-D 2800 Powder Mill Road Adelphi, MD 20783-1138	**8. PERFORMING ORGANIZATION REPORT NUMBER** ARL-TR-7532
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This report presents the results of a power utilization and packet loss study for the Extremely Lightweight Intrusion Detection (ELIDe) algorithm on an Android-based mobile device. Our results show that the hashing and inner product operations performed by the core ELIDe program alone results in negligible additional power consumption. The bulk of the excess power consumption caused by the ELIDe program is attributable to the data capturing of the Libpcap library and the internal hardware of the device that performs data capturing based on the limitation of the mobile device. Power consumption is tightly linked to the total number of network packets processed by the device, but remains tractable within the operating range of ELIDe. Results also show that ELIDe is capable of handling throughput rates of up to approximately 2.5 megabits per second (assuming a normal distribution of packet sizes) with no significant packet loss.

15. SUBJECT TERMS

ELIDe, Android, pcap

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Ken F Yu
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	UU	22	**19b. TELEPHONE NUMBER (Include area code)** (301) 394-3181

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

1. Introduction

Performing signature detection on network traffic is a computationally intensive task. Portable devices such as mobile phones, which emphasize low-power utilization over computational prowess, are traditionally ill-suited for this task.[1] Extremely lightweight intrusion detection (ELIDe) was developed as an alternative to signature detection. ELIDe breaks packets into n-grams and hashes them to produce a feature vector. Through stochastic gradient decent, a linear classifier is trained to differentiate between 2 different sets of packets.[2] Although it is missing many of the features of more complex signature-based detection software such as Snort (e.g., Transmission Control Protocol [TCP] reassembly), ELIDe maintains much of the accuracy of this standard bearer of signature-based intrusion detection while using significantly less computational resources.

ELIDe has proven to provide a level of intrusion detection through packet classification;[2] however, although it is computationally less intensive than software such as Snort, ELIDe's power utilization on a mobile device needs to be characterized. The value of ELIDe for deployment on mobile devices becomes inversely proportional to the amount of additional power needed for it to operate. ELIDe would be less useful if the runtime of a mobile device was cut in half while running this software.

2. Setup

To properly characterize the power utilization of ELIDe in regard to mobile devices, it became necessary to port the existing software to a mobile platform. Google's Android[3] was selected due to its open nature, native C++ support, and the Army's interest in the operating system (OS). To run ELIDe, the current version was ported for use on Android.[4]

2.1 Mobile Device

After ELIDe was ported to the Android mobile platform, we needed a mobile device to characterize its power utilization. A Sprint Galaxy S3 smart phone was available. The Galaxy S3 line of smart phones varies in its technical specifications depending on the carrier. Therefore, for reference, we used the Sprint Galaxy S3 with the following technical specifications:

- Qualcomm Snapdragon S4 Plus MSM8960
- Dual-core 1.5 GHz Krait processor

- Adreno 225 graphics processor
- 2048 MB of RAM
- 32 GB internal storage
- 2100 mAh battery

The ELIDe software does not make use of the built-in Adreno graphics processor in any way. Processing is handled entirely with the Krait processor. The phone was factory reset before experimentation to ensure that previously installed software would not interfere with performance and battery usage behaviors.

2.2 Network

An Android mobile phone's primary mechanism of transmitting and receiving network traffic occurs through its built-in Wi-Fi adapter. Because the typical scenario in which a mobile device uses its Wi-Fi connection is through an access point, we setup a test network consisting of an access point, laptop, and the mobile phone.

To reduce interference to the Wi-Fi connection, the laptop was connected to the access point via an Ethernet cable rather than via a wireless connection. A decrease in interference assisted in preventing packet drops and alleviated the need for the mobile device to increase the power to its radio transmissions, which prematurely drains its battery. In addition, the phone was placed in airplane mode to turn off wireless features and Wi-Fi was manually enabled. This process prevented unpredictable battery power dissipation due to wireless protocols. Furthermore, notifications—including sound and vibrations—were turned off to keep them from interfering with the battery utilization data (see Fig. 1).

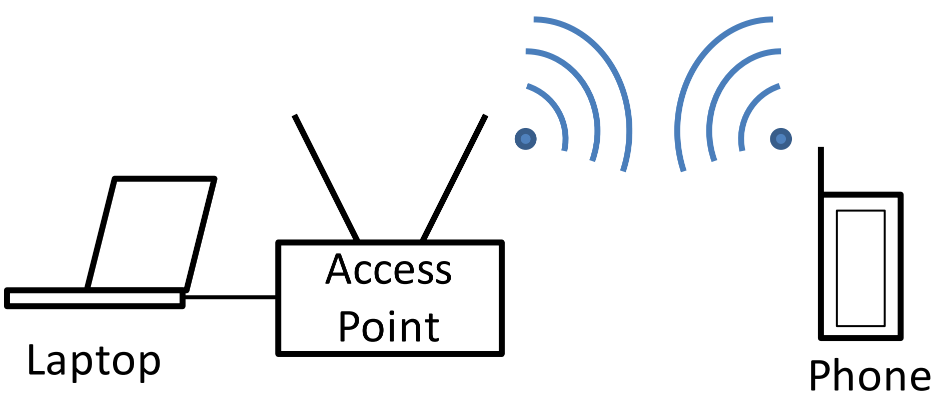

Fig. 1 Test network setup

The following devices were used alongside the mobile phone for this network:

- Cisco Wireless WAP300N
- Dell Latitude E6500 – Core 2 Duo 2.53 GHz, 4 GB of RAM
- Kali Linux 1.0.7[5]

2.3 Software Configuration

Although the ELIDe software runs on an Android device, the application is not primarily executed using the Dalvik virtual machine, but rather it is executed natively using compiled code. The majority of the application utilizes Android's Native Development Kit (NDK), which allows for the use of compiling native code including C and C++ as well as Fortran. ELIDe for Android is split into 2 main parts: the actual detection software using the NDK and the Android application front end that interfaces with the natively compiled binary that runs as a service. The Android application front end receives statistics from the service and provides an interface to change the service's configuration. The graphical front end allows users to change a number of configuration items including disabling the detection mechanisms, setting the service to save packet and power statistics to a specific file, and changing the weight file.

The software program, when intentionally started by the user, will log statistics including the number of packets classified, the number that were positively and negatively identified, and packets dropped due to the buffer being full. In addition, battery statistics can be captured. While running, the service records the current running time in seconds every time the battery storage drops a percentage, and the runtime is reset to 0 whenever the service is restarted. Using this information, we can calculate the extra power utilization that ELIDe mobile requires by finding the difference in runtime when a preset number of battery percentages drop.

2.4 Determining Power Utilization

To determine the amount of additional power required, we used the power usage statistics recorded by the Android version of ELIDe. The runtime at the final percentage recorded indicates the amount of time the device has run. If the final runtime differs slightly for ELIDe without the classification turned on than with that of ELIDe classification enabled, the change in runtime indicates that ELIDe classification does not use significantly more battery life. However, if the runtime during ELIDe classification is much smaller, this indicates that ELIDe classification requires significant amounts of power because it takes less time to run the battery down.

The percentage of additional battery power that was used can be calculated by taking the final runtime with the software configured to run ELIDe classification and the final runtime of running the software without classification. If the runtime using ELIDe classification is much lower, then the additional battery power used will be much higher.

To determine the exact percentage, the ELIDe classification runtime is divided by the runtime when ELIDe classification is disabled. Then, this number is divided by 100 to calculate the additional amount of power required for ELIDe classification.

2.5 Experimental Setup

Using the network described above, we used a laptop wired to the access point to send packets as User Datagram Protocol (UDP)[6] to the device. The payload size was adjusted by padding the payload to the size we needed. This was achieved using the “hping3”[7] utility included in Kali Linux. Libpcap[8] captures the packets, places them in a buffer, and the ELIDe classification removes a packet, determines its classification, and moves to the next packet. In the event the buffer fills, new packets are dropped until ELIDe processes another packet. Dropped packets are kept in the packet statistics.

During an experiment, while packets were sent to the phone, packet and power statistics were saved and written to the file system. To determine ELIDe’s contribution to power utilization, the same experiment was run. However, the ELIDe detection functionality was disabled. This configuration change leaves Libpcap’s functionality intact for these experiments. This configuration change was used to properly characterize the power requirements of the ELIDe process rather than ELIDe and Libpcap together. Any device looking to capture packets most likely utilizes Libpcap and thus experiences the same power utilization requirements to use the functionality.

3. Results

Before we began extensive power characterization, we tested the power utilization with ELIDe in the presence of normal background traffic produced by the phone as well as a simple Internet Control Message Protocol (ICMP) echo request with a corresponding reply from the phone. We expected a small increase in power utilization while ELIDe classified a nominal amount of traffic. For this experiment, we kept the device in operation until the battery was nearly depleted. Not only could we gage the difference in power utilization when ELIDe was classifying packets, but this process allowed us to understand the battery usage curve of the device (see Fig. 2).

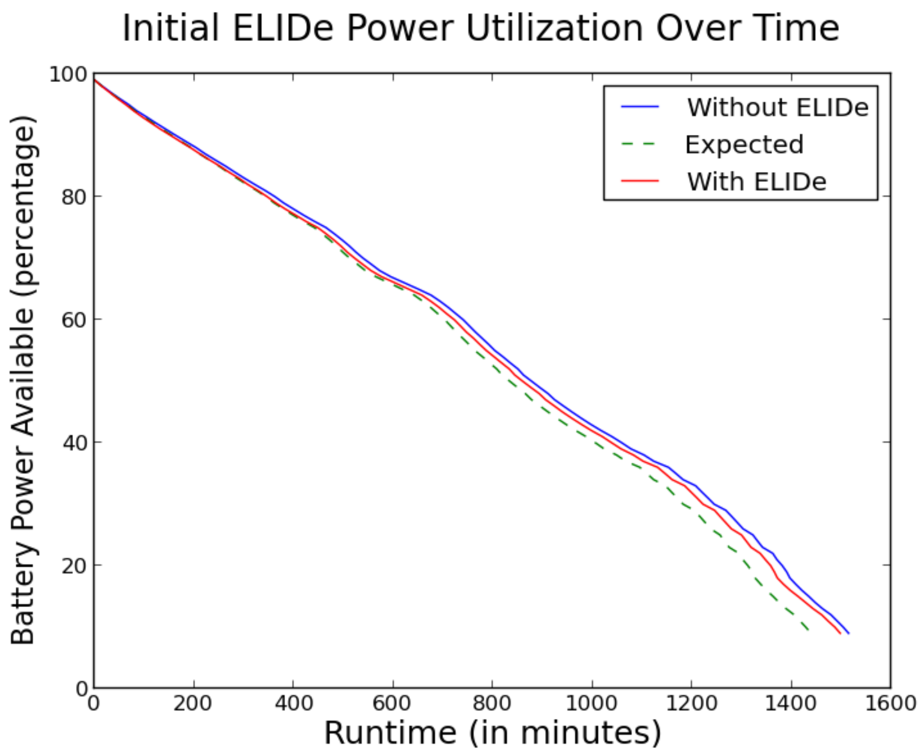

Fig. 2 Power utilization of ELIDe in the presence of nominal network traffic

Figure 2 shows the available battery life versus the runtime. As expected, while the device continues to run, the available power reported by the OS declines. Based on the preliminary results from the original development of the ELIDe approach, we expected ELIDe detection to require approximately 5% in the presence of nominal amounts of traffic—the actual power utilization was much lower. ELIDe only required an additional 1.1% in running power to perform its functions in the presence of nominal traffic.

The major conclusion we draw from this result is that ELIDe does not significantly impact the battery utilization of the device in the presence of nominal amounts of network traffic. This indicates that the ELIDe service could continue to run in the background during periods of network inactivity with little impact on battery life.

3.1 Libpcap Overhead

The purpose of this report is to document the power utilization of ELIDe on an Android device, due to its dependence on Libpcap to capture network traffic. Therefore, we briefly investigated how Libpcap usage may affect power utilization independently. We performed an experiment in which we sent network packets with 600-byte padding at 1 Mbp to the mobile device and did not perform any classification. Libpcap captured the packets and placed them in a buffer. The effect

of simply capturing packets and placing them in the buffer does not significantly affect battery life (see Fig. 3).

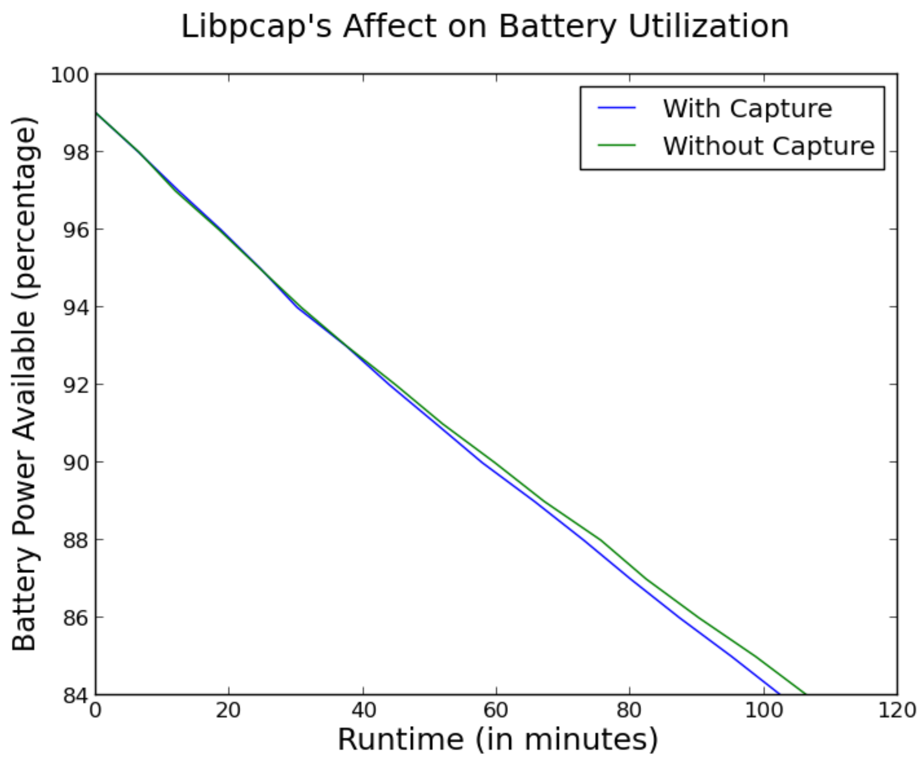

Fig. 3 Libpcap packet capture battery utilization

Performing packet capture alone uses approximately 3% of battery life in this particular situation. Although we could have performed an in-depth analysis on how Libpcap[8] affected battery life, the purpose of this research was to investigate ELIDe power utilization. Libpcap was used in all experiments even with classification turned off, to determine the power consumption of ELIDe itself rather than the entire process of packet inspection.

3.2 Throughput Changes

To determine ELIDe usefulness as a mobile intrusion detection system, we must measure how much additional drain occurs on the battery when running on a mobile device. Nominal network traffic did not significantly impact the phone's power. However, phones experience spikes in traffic or sustain data transmissions, pushing throughput to much higher than nominal rates. We looked at the effect on battery utilization in regard to different rates of data being sent to the mobile device.

To test the throughput, we used hping3 on the Kali Linux distribution installed on the laptop. We sent packets with an arbitrary payload of 600 bytes at varying speeds

to the mobile device. This was achieved by varying the rate at which a packet was sent from the laptop. We expected power utilization to increase as the data rate to the device increased because ELIDe would need to process more packets in the same amount of time—causing it to drain the battery faster to perform the equivalent amount of processing in a shorter time. We see that lower data rates between throughput and power utilization are not linear (see Fig.4).

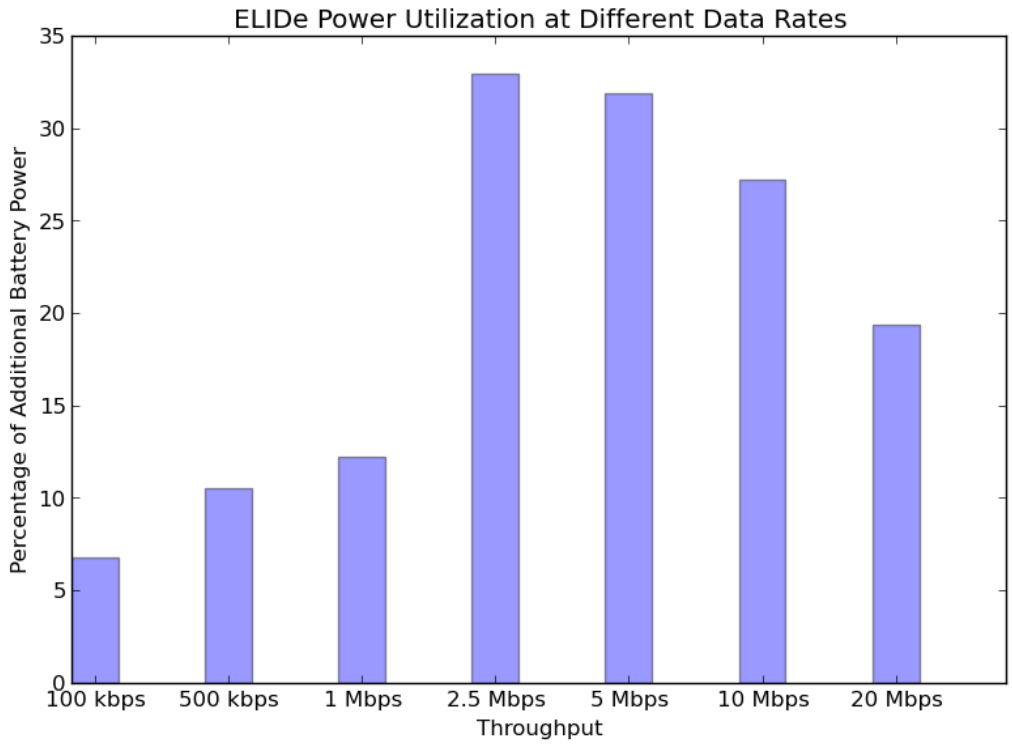

Fig. 4 Power utilization at different throughputs

As throughput increases, the power utilization increases until speed reaches approximately 1 Mbp. ELIDe classification requires 12.2% of additional power compared to when ELIDe classification is disabled.

Once at 2.5 Mbps, the relationship changes drastically with ELIDe requiring an additional 31.9% of power to process this rate. In addition, as the rate continues to increase, the utilization actually starts to drop. This behavior was unexpected because more packets would require additional power for the processing needed for classification. We believe that the drop in power utilization could be due to ELIDe dropping packets. The processor in the device may not have been fast enough to keep up with the rate of packets, filling the buffer, and causing ELIDe to drop packets.

3.3 Dropped Packets

We examined the packet statistics recorded by ELIDe to determine whether there was an association between decreased power utilization at higher throughputs. When ELIDe classification is disabled, even at higher throughputs, there is little-to-no packet loss (see Fig. 5).

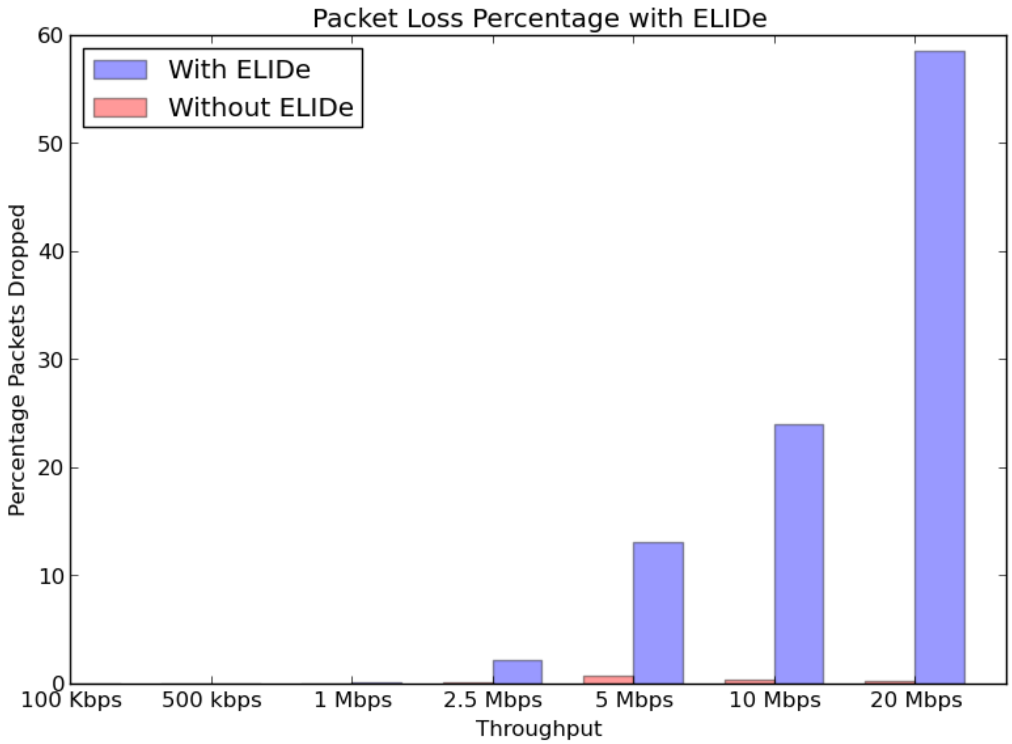

Fig. 5 ELIDe packet loss at various data rates

This result also applies to ELIDe classification for throughputs under 2.5 Mbps when a significant amount of packets is dropped (not classified). After 2.5 Mbps, the percentage of dropped packets increases almost exponentially. There is a distinct relationship between the lower power utilization at higher throughput; after 2.5 Mbps, power utilization drops as throughput increases. This relationship seems counterintuitive, and we have found that a relationship also exists between the lower power utilization at higher throughputs and the amount of dropped packets (see Fig. 6). Packets that are received when the Libpcap[8] buffer is full are not saved for ELIDe classification.

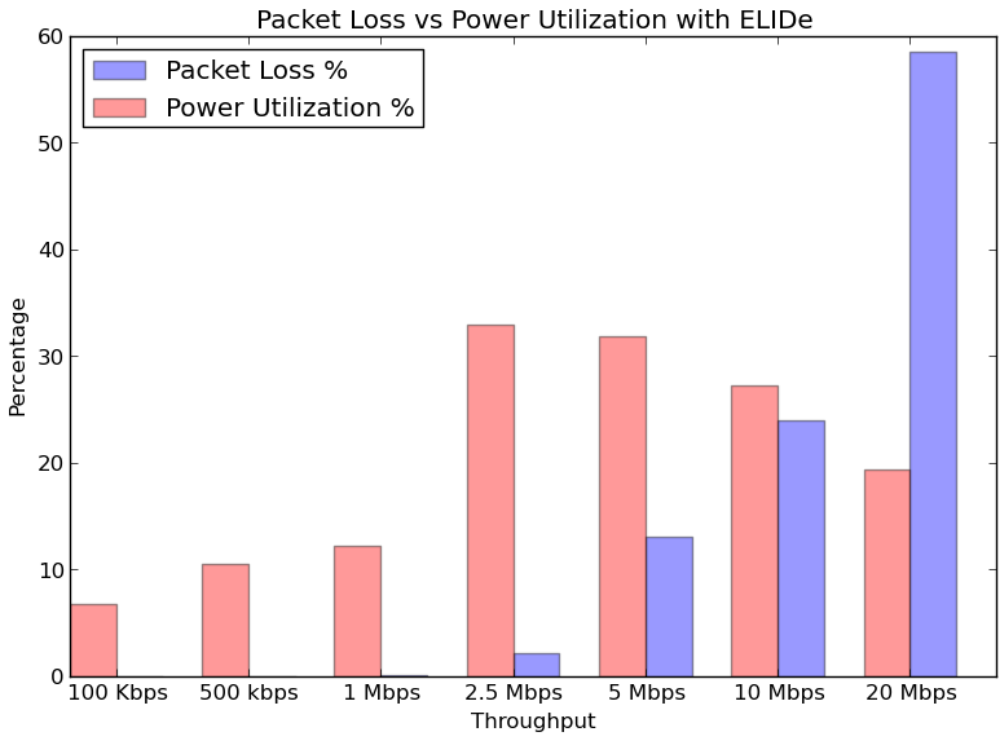

Fig. 6 Power utilization vs. packet loss at different throughputs

Although it appears that ELIDe processes higher throughput rates at lower power, it is doing so at the cost of dropping packets from classification—indicating that if the intent is to classify as much network traffic as possible, then the device's throughput should be limited to lower speeds. However, if a certain percentage of unclassified packets are allowed, then higher throughputs can be used on the mobile device. However, dropped packets are not classified, and, therefore, this allows malicious payloads to go undetected.

3.4 Varying Payload Sizes

After we established that ELIDe can be used for various levels of the throughput, we wanted to determine whether the size of the packet affects the power utilization. To do this, we used hping3 and adjusted the payload to varying sizes. For each payload size, a corresponding rate was used so that we would maintain 1 Mbp throughput. We found that the power utilization can vary wildly depending on the payload size (see Fig. 7).

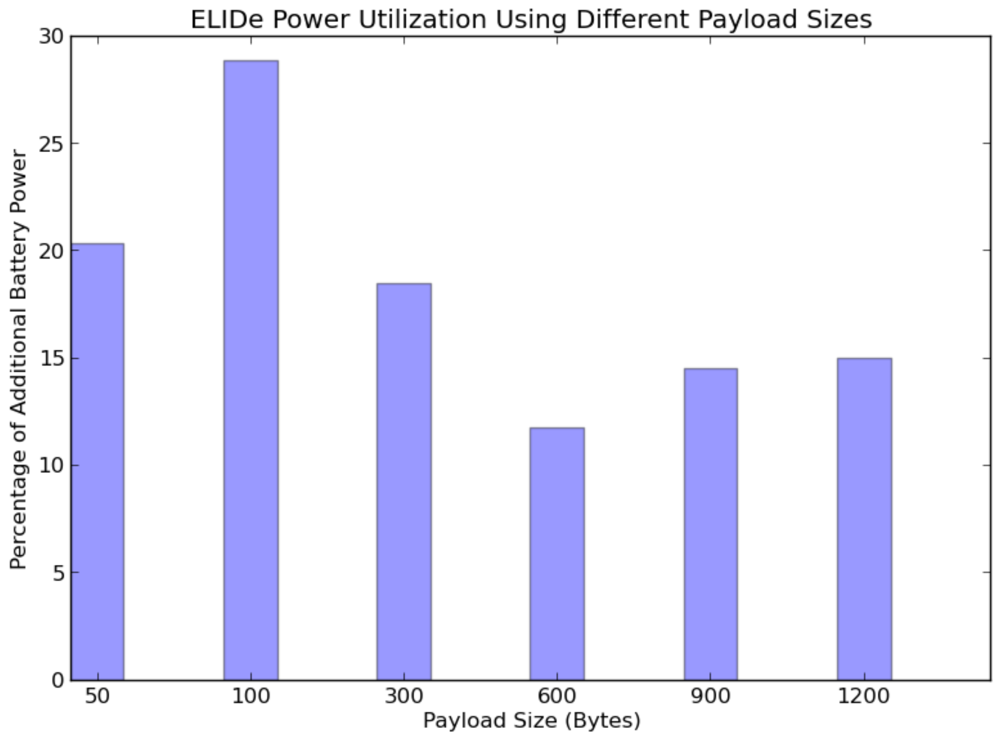

Fig. 7 Power utilization using different payload sizes

Figure 8 shows that at 100 bytes, the power utilization is at its largest, dropping significantly at 300 bytes and 600 bytes, but increasing slowly as the payload increases to 1200 bytes. It appears that the ideal level for ELIDe classification at 1 Mbps is a 600-byte payload. ELIDe requires more power to process smaller payloads because this means that several additional packets can be sent at the same speeds as compared with the larger packets. This indicates that ELIDe expends most of its energy processing each packet as opposed to the amount of data that it receives. However, the outlier that violates this pattern is 50-byte payloads.

3.5 Dropped Packets

Considering that more power, and therefore more processing power, is required for a smaller payload size, it stands to reason that the throughput utilizing the smallest payload, 50 bytes, would require the most power. The fact that it dropped when it should be higher is similar to the same drop in utilization at higher bandwidths. The drop in power utilization could be caused by dropped packets. We found that there is a relationship between dropped packets and inexplicable drops in power utilization (see Fig. 8).

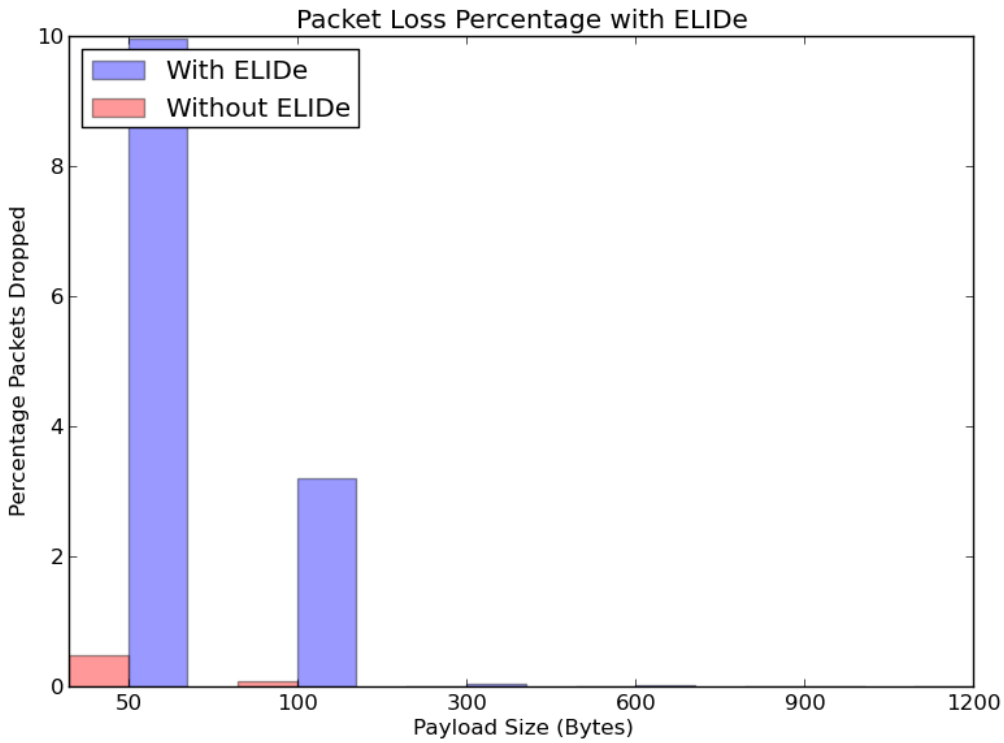

Fig. 8 Packet loss measured at varying payload sizes

In this case, a rate of 1 Mbps using 50-byte payloads had 10% of the collected packets dropped. However, we found that the relationship clearly establishes that ELIDe has more difficulty classifying smaller packets and may drop packets before they can be classified to keep up with rate (see Fig. 9).

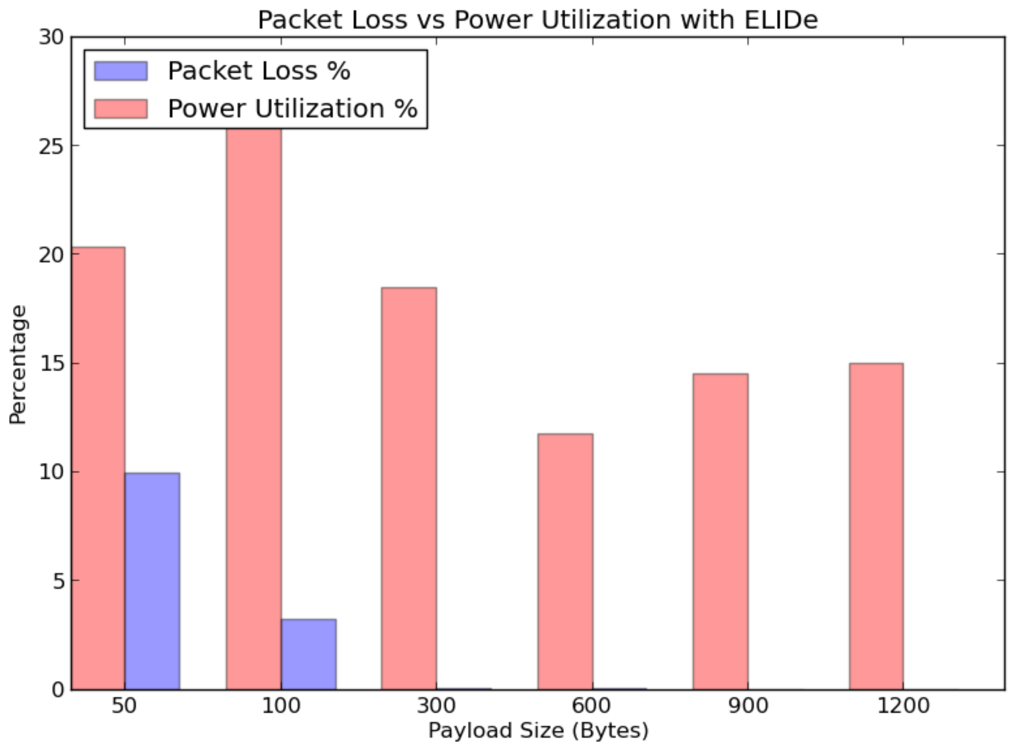

Fig. 9 Comparison of packet loss vs. power utilization

Although it is possible to throttle bandwidth, it is difficult (almost impossible) to change the shapes of packets received by a mobile device. Environments that make greater utilization of smaller packet sizes would need to adjust their bandwidth to lower levels to compensate.

4. Conclusions

ELIDe on Android can perform packet classification on a standard mobile device effectively on bandwidth-constrained networks. As the throughput increases to the device, the power utilization increases to a threshold where ELIDe can no longer process packets fast enough to keep the buffer from filling. Once the buffer is full, packets are dropped, with a corresponding drop in power utilization.

We found that packet count, not the overall amount of data sent to the device, affects its power utilization. While keeping the throughput constant, traffic made up of smaller payloads requires more power than those that use larger sizes. Similar to high throughput, ELIDe drops packets at smaller payload sizes, causing a corresponding drop in power utilization.

A typical network does not have the same continuous bandwidth and packet sizes. They can vary heavily depending on the situation or types of data traversing the

network. For ELIDe to be used effectively, those who deploy this capability to a network must be able to understand its characteristics. Networks using smaller packets at high rates may need to tolerate lost packets that evade classification, or bandwidth may need to be throttled.

5. References

1. Hugelshofer F, Smith P, Hutchison D, Race NJ. OpenLIDS: A lightweight intrusion detection system for wireless mesh networks. In Proceedings of the 15th annual International Conference on Mobile Computing and Networking, 2009.

2. Chan, RJ, Harang RE, Payer GS. Extremely Lightweight Intrusion Detection (ELIDe). Adelphi Laboratory Center (MD): Army Research Laboratory (US), 2013. Report No.: ARL-CR-0730. Also available at http://www.arl.army.mil/www/default.cfm?technical_report=6990.

3. Rogers R, Lombardo J, Mednieks Z, Meike B. Programming with the Google SDK. O'Reilly Media, Inc, 2009.

4. Yu KF, Payer GS. Porting Extremely Lightweight Intrusion Detection (ELIDe) to Android. Adelphi Laboratory Center (MD): Army Research Laboratory (US), September 2015. Report No.: ARL-TN-0681.

5. Kali Linux. [accessed 2014 Sept 30]. http://www.kali.org/.

6. Postel, J. User datagram protocol. Isi, 1980.

7. Hping3. [accessed 2014 Sept 30]. http://www.hping.org/hping3.html.

8. TCPDump and LibPCAP. [accessed 2014 March 13]. http://www.tcpdump.org/.

1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
2 (PDF)	DIRECTOR US ARMY RSRCH LAB RDRL CIO LL IMAL HRA MAIL & RECORDS MGMT
1 (PDF)	GOVT PRINTG OFC A MALHOTRA
1 (PDF)	DIRECTOR US ARMY RSRCH LAB RDRL CIN D K YU

INTENTIONALLY LEFT BLANK.